Exploratory Statistical Data Analysis With R Software (ESDAR)

Swayam Prabha

Lecture 12

Frequency Distribution with R Software

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

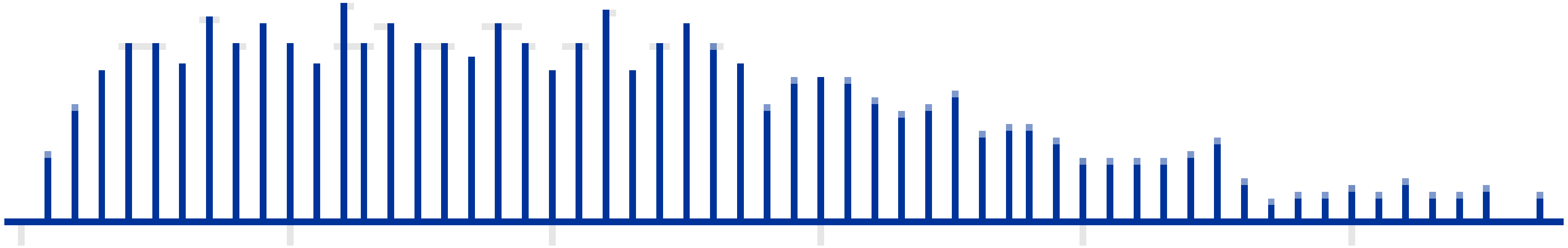
Slides can be downloaded from

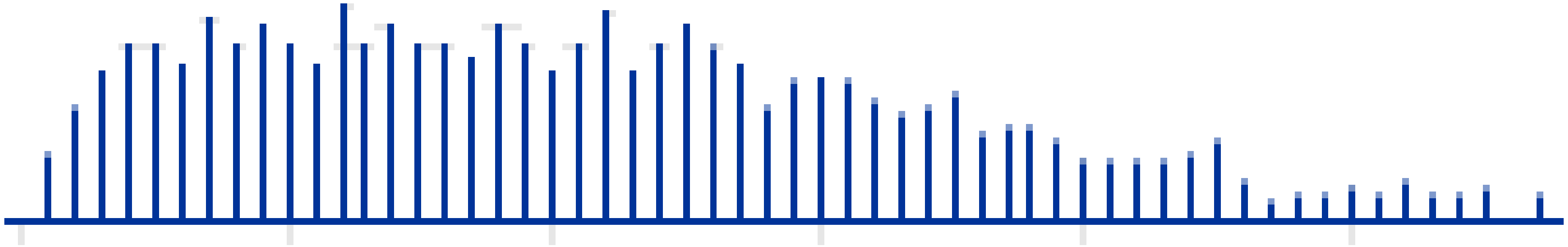
http://home.iitk.ac.in/~shalab/sp

Frequency Polygon and Frequency Curve

- **Obtain the mid points of class intervals.**
- **Mark frequency on y-axis against the midpoints.**
- **Join the frequency points of all the rectangles by straight lines.**
- **Join the end points on *x*-axis.**
- **This is frequency polygon.**
- **Joining the top midpoints of all rectangles by a smooth curve using a smooth hand, without stopping the pen.**
- **This is frequency curve.**

Frequency Polygon and Frequency Curve





Cumulative Frequency Curve

- Obtain the mid points of class intervals.
- Mark cumulative frequency on y-axis against the midpoints.
- Join the cumulative frequency points of all the rectangles by straight lines.
- This is cumulative frequency curve.

Frequency Distribution

First step is to find the range of the data values which can be partitioned into class interval.

Use command `range` which returns a vector containing the minimum and maximum of all the given arguments.

Usage:

`range(data vector)` returns a vector containing the minimum and maximum of all the given arguments.

Frequency Distribution

Example:

Following are the time taken (in seconds) by 20 participants in a race.

32, 35, 45, 83, 74, 55, 68, 38, 35, 55, 66, 65, 42, 68, 72, 84, 67, 36, 42, 58.

The data is summarized in class intervals

31-40, 41-50, 51-60, 61-70, 71-80 and 81-90

Frequency Distribution

Example (contd.):

Class intervals	Mid point	Absolute frequency (or frequency)	Relative Frequency	Cumulative Frequency
31 – 40	35.5	5	5/20 = 0.25	5
41 – 50	45.5	3	3/20 = 0.15	5+3 = 8
51 – 60	55.5	3	3/20 = 0.15	5+3+3 = 11
61 – 70	65.5	5	5/20 = 0.25	5+3+3+5 = 16
71 – 80	75.5	2	2/20 = 0.01	5+3+3+5+2 = 18
81 - 90	85.5	2	2/20 = 0.01	5+3+3+5+2+2 = 20
	Total	20	1	

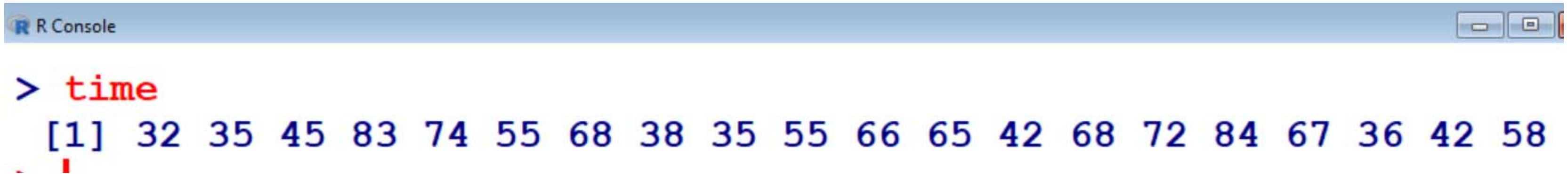
Frequency Distribution

Example (contd.):

```
> time

[1] 32 35 45 83 74 55 68 38 35 55 66 65 42 68

72 84 67 36 42 58
```



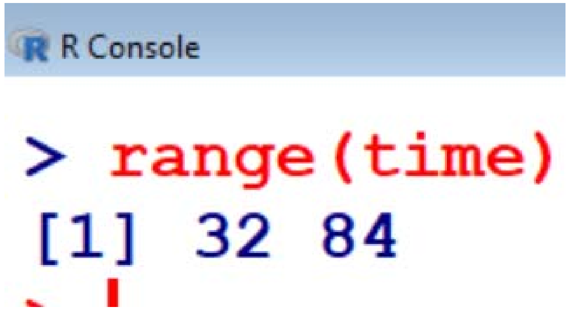
```
> time
 [1] 32 35 45 83 74 55 68 38 35 55 66 65 42 68 72 84 67 36 42 58
```

Frequency Distribution

Example (contd.):

```
> range(time)

[1] 32 84
```



```
> range(time)
[1] 32 84
```

This result gives an information and it looks reasonable to divide the data in class following intervals:

31-40,   41-50,   51-60,   61-70,   71-80 and   81-90

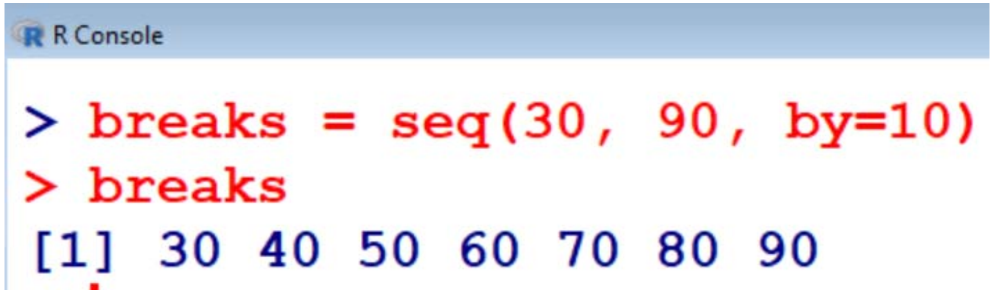
Create a sequence starting from 30 to 90 at an interval of 10 integers denoting the width.

Frequency Distribution

Example (contd.):

Create a sequence starting from 30 to 90 at an interval of 10 integers denoting the width.

```
breaks = seq(30, 90, by=10)  # sequence at
                                interval of 10 integers
> breaks = seq(30, 90, by=10)
> breaks
[1] 30 40 50 60 70 80 90
```



```
> breaks = seq(30, 90, by=10)
> breaks
[1] 30 40 50 60 70 80 90
```

Frequency Distribution

Now we need to convert Numeric to Factor using a command `cut`

Usage: `cut(data vector, breaks, right = FALSE)` divides the range of `data vector` into intervals and codes the values in `data vector` according to which interval they fall.

`breaks` is a numeric vector of two or more unique cut points or a single number (greater than or equal to 2) giving the number of intervals into which `data vector` is to be cut.

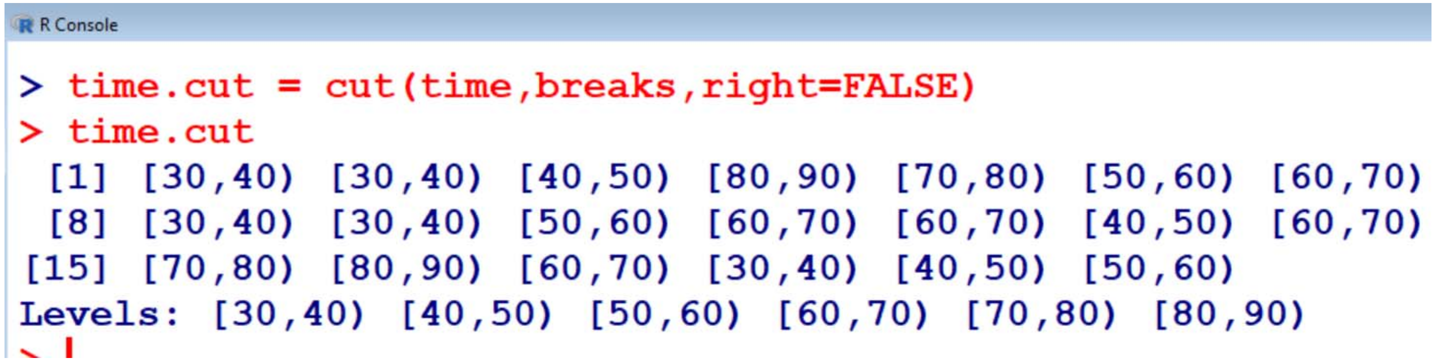
As the intervals are to be closed on the left, and open on the `right`, we set the right argument as `FALSE`.

Frequency Distribution

Example (contd.):

Now we classify the time data according to the width intervals with `cut`.

```
> time.cut = cut(time,breaks,right=FALSE)
> time.cut
 [1] [30,40) [30,40) [40,50) [80,90) [70,80) [50,60) [60,70)
 [8] [30,40) [30,40) [50,60) [60,70) [60,70) [40,50) [60,70)
[15] [70,80) [80,90) [60,70) [30,40) [40,50) [50,60)
Levels: [30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
```

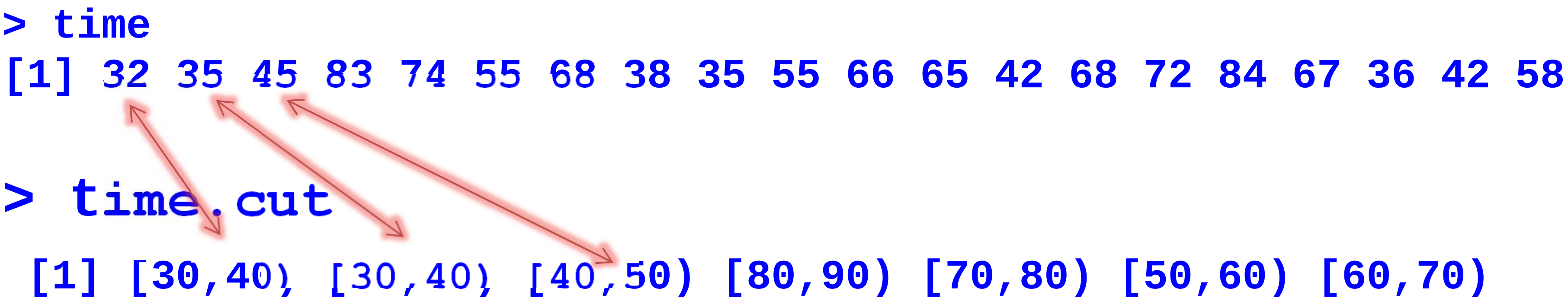


```
> time.cut = cut(time,breaks,right=FALSE)
> time.cut
 [1] [30,40) [30,40) [40,50) [80,90) [70,80) [50,60) [60,70)
 [8] [30,40) [30,40) [50,60) [60,70) [60,70) [40,50) [60,70)
[15] [70,80) [80,90) [60,70) [30,40) [40,50) [50,60)
Levels: [30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
> |
```

Frequency Distribution

Example (contd.):

Interpretation of outcome. Recall



```
> time
[1] 32 35 45 83 74 55 68 38 35 55 66 65 42 68 72 84 67 36 42 58

> time.cut
 [1] [30,40) [30,40) [40,50) [80,90) [70,80) [50,60) [60,70)
 [8] [30,40) [30,40) [50,60) [60,70) [60,70) [40,50) [60,70)
[15] [70,80) [80,90) [60,70) [30,40) [40,50) [50,60)
Levels: [30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
```

Frequency Distribution

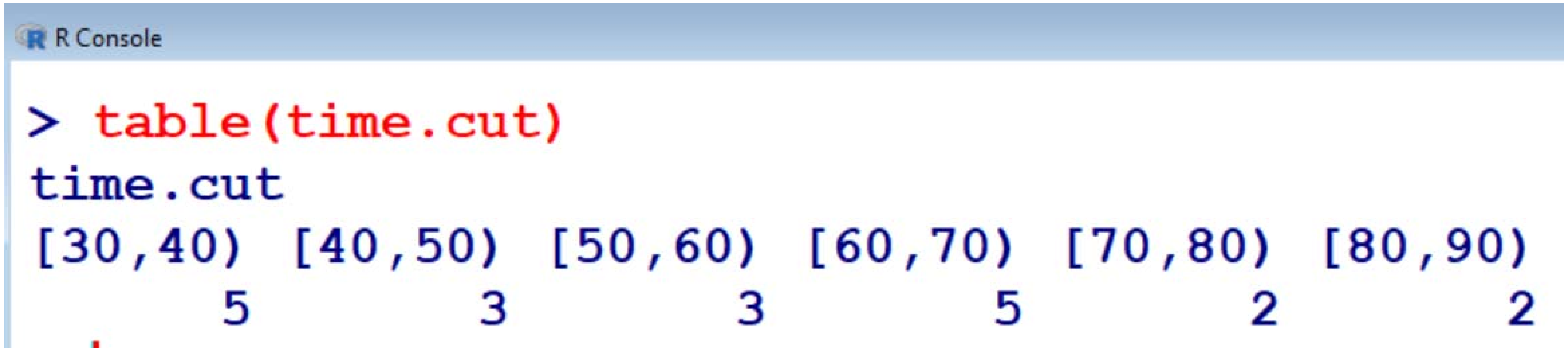
Now we can compute the absolute frequency of time data in each width interval with the `table` function

`table(variable)` creates the absolute frequency of the `variable` of the data file which generates the frequency distribution of the data on `variable`.

Frequency Distribution

Example (contd.):

```
> table(time.cut)
time.cut
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
      5       3       3       5       2       2
```



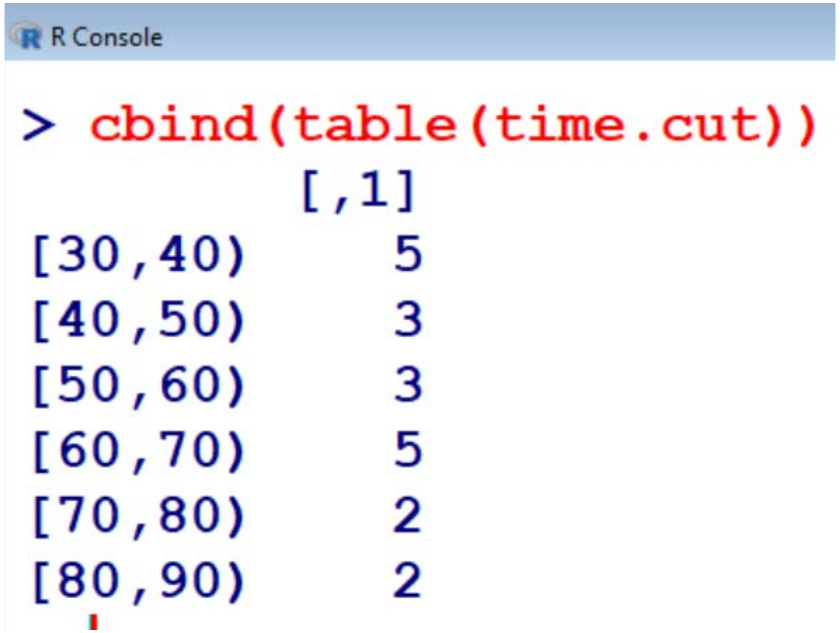
```
> table(time.cut)
time.cut
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
      5       3       3       5       2       2
```

Frequency Distribution

Use the cbind function to print the frequency distribution in column format.

Example (contd.):

```
> cbind(table(time.cut))
        [,1]
[30,40)    5
[40,50)    3
[50,60)    3
[60,70)    5
[70,80)    2
[80,90)    2
```



```
> cbind(table(time.cut))
        [,1]
[30,40)    5
[40,50)    3
[50,60)    3
[60,70)    5
[70,80)    2
[80,90)    2
```

Frequency Distribution

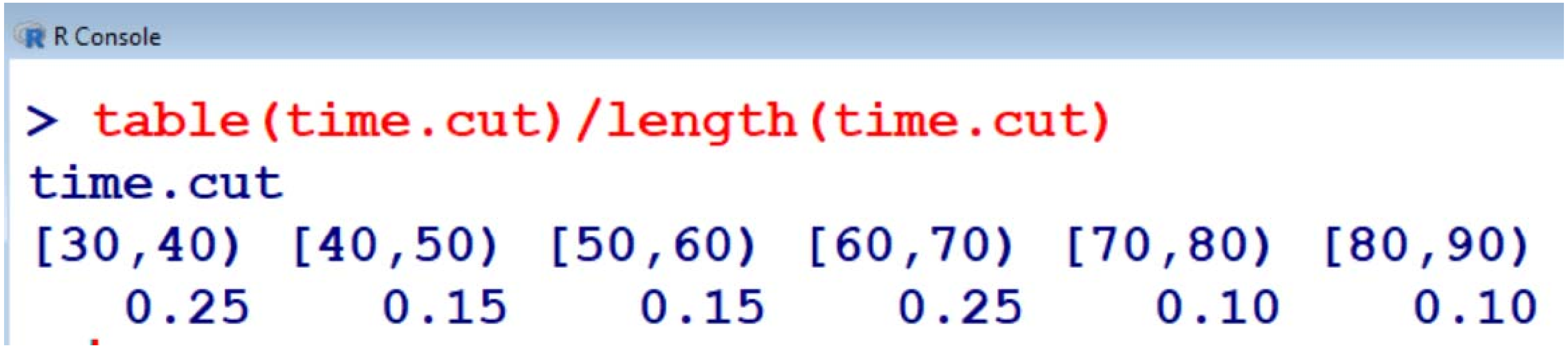
To compute the relative frequency of time data in each width interval with the `table` function with `length` function

`table(variable)/length(variable)` creates the relative frequency of the `variable` of the data file which generates the frequency distribution of the data on `variable`.

Frequency Distribution

Example (contd.):

```
> table(time.cut)/length(time.cut)
time.cut
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
   0.25    0.15    0.15    0.25    0.10    0.10
```



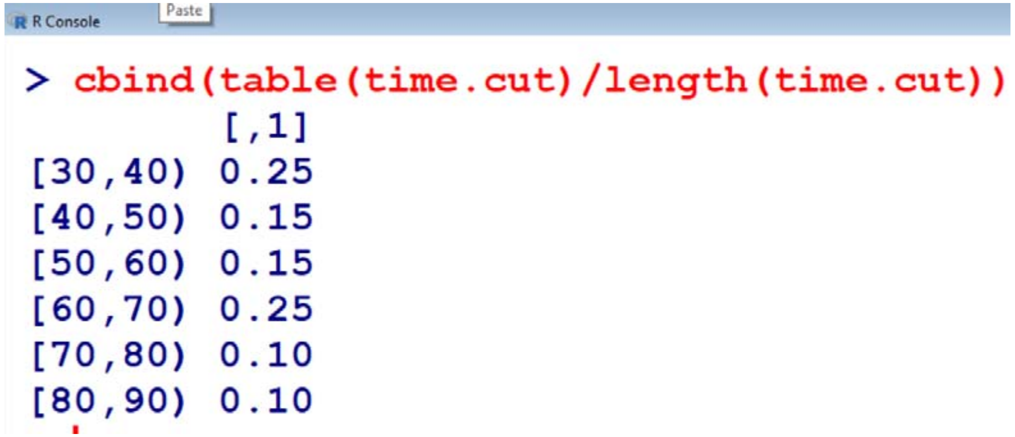
```
> table(time.cut)/length(time.cut)
time.cut
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
   0.25    0.15    0.15    0.25    0.10    0.10
```

Frequency Distribution

Use the `cbind` function to print the frequency distribution in column format.

Example (contd.):

```
> cbind(table(time.cut)/length(time.cut))
         [,1]
[30,40) 0.25
[40,50) 0.15
[50,60) 0.15
[60,70) 0.25
[70,80) 0.10
[80,90) 0.10
```



```
> cbind(table(time.cut)/length(time.cut))
         [,1]
[30,40) 0.25
[40,50) 0.15
[50,60) 0.15
[60,70) 0.25
[70,80) 0.10
[80,90) 0.10
```

Cumulative Distribution Function (CDF) for data

It gives us an idea about the cumulative frequencies up to a certain point.

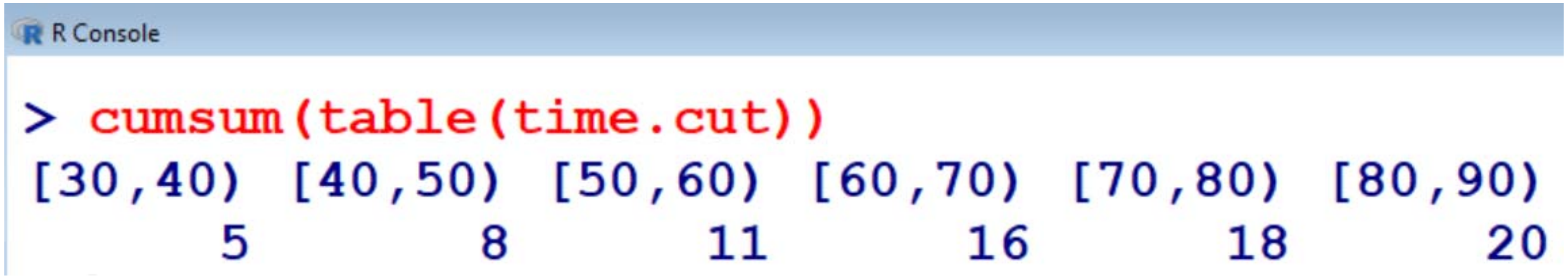
The cumulative frequencies are computed by the function `cumsum`

Usage: `cumsum(table(variable))` returns a vector whose elements are the cumulative sums of the elements of the frequencies in the `variable` in the argument.

Cumulative Distribution Function (CDF) for data

Example (contd.):

```
> cumsum(table(time.cut))
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
      5       8      11      16      18      20
```



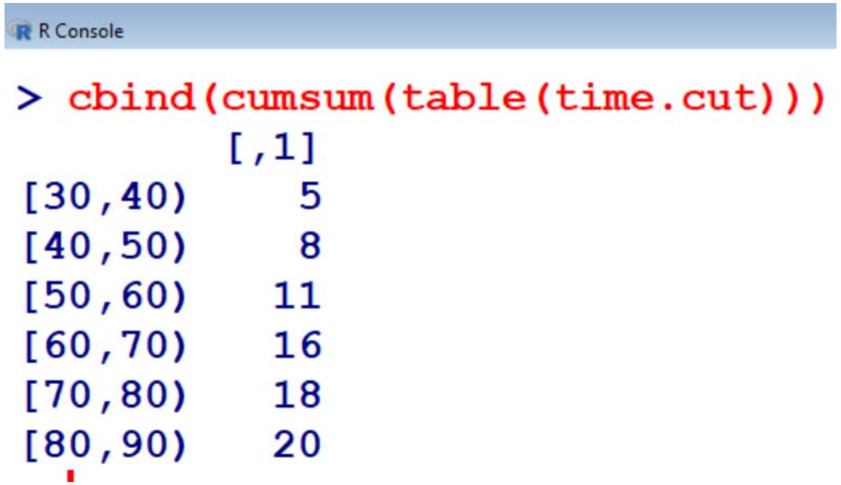
```
> cumsum(table(time.cut))
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
      5       8      11      16      18      20
```

Cumulative Distribution Function (CDF) for data

Use the `cbind` function to print the cumulative frequency distribution in column format.

Example (contd.):

```
> cbind(cumsum(table(time.cut)))
        [,1]
[30,40)    5
[40,50)    8
[50,60)   11
[60,70)   16
[70,80)   18
[80,90)   20
```



```
> cbind(cumsum(table(time.cut)))
        [,1]
[30,40)    5
[40,50)    8
[50,60)   11
[60,70)   16
[70,80)   18
[80,90)   20
```

Cumulative Distribution Function (CDF) for data

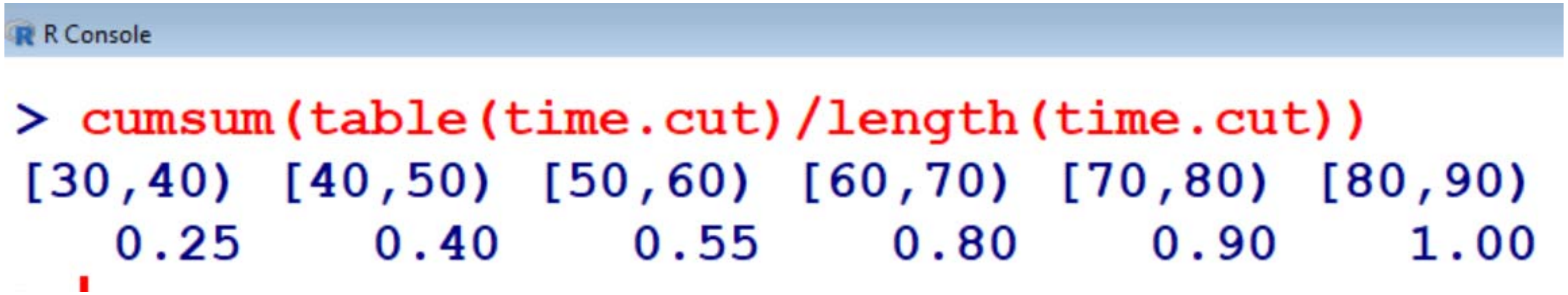
If the cumulative frequencies are to be computed based on relative frequency then the function `cumsum` is used with `table(variable)/length(variable)`

Usage: `cumsum(table(variable)/length(variable))` returns a vector whose elements are the cumulative sums of the elements of the relative frequencies in the `variable` in the argument.

Cumulative Distribution Function (CDF) for data

Example (contd.):

```
> cumsum(table(time.cut)/length(time.cut))
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
   0.25    0.40    0.55    0.80    0.90    1.00
```



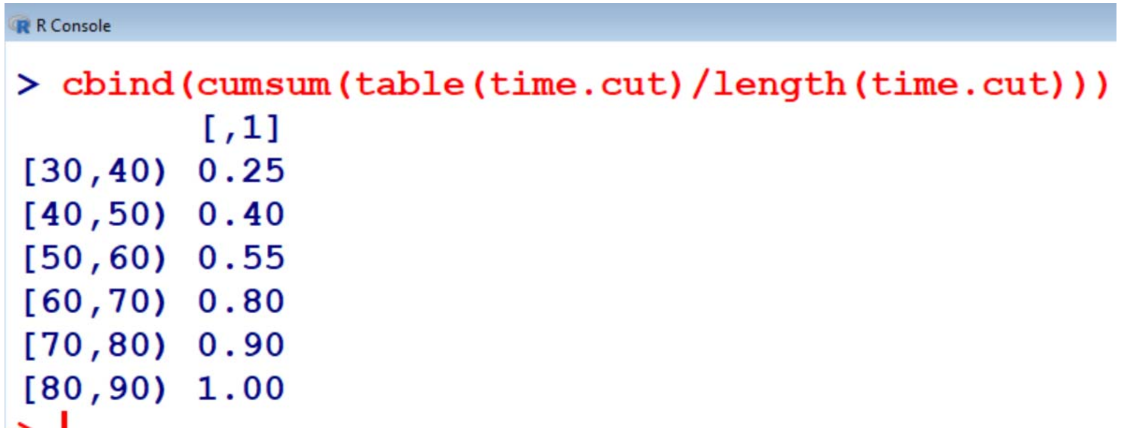
```
> cumsum(table(time.cut)/length(time.cut))
[30,40) [40,50) [50,60) [60,70) [70,80) [80,90)
   0.25    0.40    0.55    0.80    0.90    1.00
```

Cumulative Distribution Function (CDF) for data

Use the `cbind` function to print the cumulative relative frequency distribution in column format.

Example (contd.):

```
> cbind(cumsum(table(time.cut)/length(time.cut)))
        [,1]
[30,40) 0.25
[40,50) 0.40
[50,60) 0.55
[60,70) 0.80
[70,80) 0.90
[80,90) 1.00
```



```
> cbind(cumsum(table(time.cut)/length(time.cut)))
        [,1]
[30,40) 0.25
[40,50) 0.40
[50,60) 0.55
[60,70) 0.80
[70,80) 0.90
[80,90) 1.00
```